necessarily result in high indoor Rn concentration. Depending on the time of observations in a year and fluctuations in the meteorological parameters, radon entry into the house and the Rn concentrations in the house vary. During meteorologically stable conditions, it is possible to encounter more than high Rn concentrations in the houses situated in the neighbourhood of the areas containing more than normal radium content in the soil. For a proper interpretation of the results obtained from the measurements of indoor Rn concentrations in the high background areas, it appears that one should study the seasonal variations of indoor Rn levels in these areas and relate them with meteorological parameters. Also it is necessary to make a study of the radon entry into the house from the soil via cracks and joints in the floor and loose pipe fittings through the basements and leakages at the wall-floor joints.

1. Mishra, U. C., *A study of natural and fallout radioactivity in Indian soils by gamma spectrometry*, Ph. D, thesis, University of Bombay, 1970.
2. Mishra, U. C., *Natural and fallout Gamma Nuclides in Indian soils*, Natural Radiation Environment-II, CONF–720805–P1, 1972, pp. 333–345.
3. Ramachandran, T. V. and Mishra, U. C., *Geophys. Res. Bull.*, 1989, **27**, 41.
4. Mishra, U. C. and Subba Ramu, M. C., *Bull. Radiat. Prot.*, 1989, **12**, 1.
5. Subba Ramu, M. C., Ramachandran, T. V., Muraleedharan, T. S. and Shaikh, A. N., Measurements of indoor randon working levels in normal natural background regions in India using passive detectors. Paper presented in 8th National Symposium on Radiation Physics, 1990, Jan 17–19, Bombay.
6. *Lung Cancer Risk from Indoor Exposures to Radon Daughters* International Commission on Radiation Protection Publication 50, 1986, pp. 14–17.
7. Subba Ramu, M. C., Shaikh, A. N., Muraleedharan, T. S. and Ramachandran, T. V., *Sci. Total Environ.*, 1990 (in press).
8. Subba Ramu, M. C., Shaikh, G. N., Muraleedharan, T. S. and Ramachandran, T. V., *Bull. Radiat. Prot.*, 1988, **11**, 81.
9. Pearson, J. E., Final Report, Dept. of Gen. Engineering, University of Illinois, Urbana, 1965.

12 March 1990

Performance characteristics of a hypercube-type parallel computer

K. Neelakantan, P. P. Ghosh, M. S. Ganagi, G. Athithan, M. V. Atre and G. Venkataraman
Advanced Numerical Research and Analysis Group (ANURAG), P. O. Kanchanbagh, Hyderabad 500 258, India

We present a status report on a project to build a hypercube-type parallel computer. LINPACK benchmark results for three pilot models are reported, and comparisons made with a similar system made by Intel Corporation. The influence of interprocessor communication speed on the LINPACK ratings achievable is discussed. Plans in progress to achieve the target system are also briefly outlined.

PROJECT PACE has the objective of designing, developing and building a high-speed concurrent (parallel) computer. Funded by the Defence Research and Development Organization (DRDO) of the Government of India, the project aims at meeting the needs of our fluid dynamicists. Computational fluid dynamics (CFD) is among the most demanding of calculations performed on computers, a typical one calling for 10^{12} arithmetical floating-point operations in about 10^4 seconds, implying a speed of 10^8 floating-point operations per second or 100 megaflops (MFLOPS). Moreover, such speeds must be available with double-precision arithmetic, and for FORTRAN programs. Notwithstanding the primary objective of our project, we believe that our system would be useful for a number of other scientific applications as well, which is the reason for this report.

System architecture

Historically the standard route to heavy number crunching has been via the so-called supercomputer, of which the famous CRAY is the canonical example. While a technical marvel, supercomputers require very sophisticated technologies for their manufacture. Fortunately, thanks to advances in microprocessor technology, the notion of parallel computing long

advocated by computer scientists can now be seriously considered as an alternative solution to computationally intense problems.

Among the many architectures possible for parallel computers, we have favoured one related to the hypercube pioneered at Caltech and evolved through various versions[1, 2]. A d-dimensional hypercube has 2^d nodes or lattice sites. While traditionally it is the practice to have only one processing element (PE) at each node (rather like having a single atom at each site of a Bravais lattice), one could also decorate each corner of the hypercube with a *cluster* of PEs (rather like attaching a cluster of atoms to each site of a Bravais lattice). Our project calls for a 128-processor system, which has been named PACE-128 (ref. 3). Initially we planned to configure it as a 7-dimensional hypercube. After examining various technical aspects, it has been decided to configure PACE-128 as a 4-dimensional hypercube, with a cluster of 8 PEs at each lattice site. The PEs belonging to a cluster are on a VME bus. Though physically each cluster has a bus architecture, it is treated logically as a 3-dimensional hypercube. In short, our revised version is a 4-dimensional hypercube with a (logical) 3-dimensional hypercube at each lattice site.

Systems presently investigated

The experiments reported here have been carried out on three pilot models, particulars of which are given in Table 1. Microprocessors of the Motorola 680X0 family have been chosen for the PE, floating-point acceleration being achieved through the use of the Motorola coprocessors 68881 and 68882. The objectives of the present experiments were (i) to gain experience with parallel architecture, (ii) to study how throughput is influenced by the ratio of communication time to computation time, and (iii) to develop suitable system software. The details of our design philosophy and our approach to the parallelization of application software are described elsewhere[4]. The hypercube is managed by a front-end processor (FEP), also based on 680X0. The FEP operating system is UNIX V.3*, and the characteristics of the system software are outlined in Figure 1.

Two types of communications are needed in our hypercube: (i) from FEP to node and (ii) node to node. In PACE-4 MK I, communications were handled by ETHERNET† (CSMA/CD communication scheme based on IEEE 802.3 protocol), while PACE-4 MK II and PACE-8 use VME backplane. Our system software is such that, from a programmer's point of view, the system can be regarded as a hypercube. This is on account of the complete connectivity that exists on the VME bus. The connectivity of a 3-dimensional hypercube is a subset of the completely connected case.

Table 1. Brief particulars of the systems studied.

	PACE-4 MK I	PACE-4 MK II	PACE-8 MK I
Front-end processor			
CPU	68020/68881	68020/68881	68030/68882
Clock	16 MHz	16 MHz	25 MHz
OS/compiler	UNIX V.3/ Softek	UNIX V.3/ Softek	UNIX V.3/ Greenhills
Memory	3 MB	1 MB	4 MB
Winchester DD	80 MB	80 MB	2 × 170 MB
Cartridge DD	60 MB	60 MB	150 MB
Floppy DD	1.2 MB	1.2 MB	1.2 MB
Ports	6 Serial, 1 parallel	6 Serial, 1 parallel	6 Serial, 1 parallel
Node			
CPU	68020	68020	68030
Coprocessor	68881	68881	68882
Clock	16 MHz	16 MHz	25 MHz
Memory per node	1 MB	1 MB	4 MB
Node software	Monitor	PSOS[††]	PSOS[††]
Communication	Ethernet	VME	VME
Communication speed (packet size 100)	~30 kbits/sec	0.9 Mbits/sec	4.92 Mbits/sec

Speedup

The speedup S of a parallel computer is defined as[5]

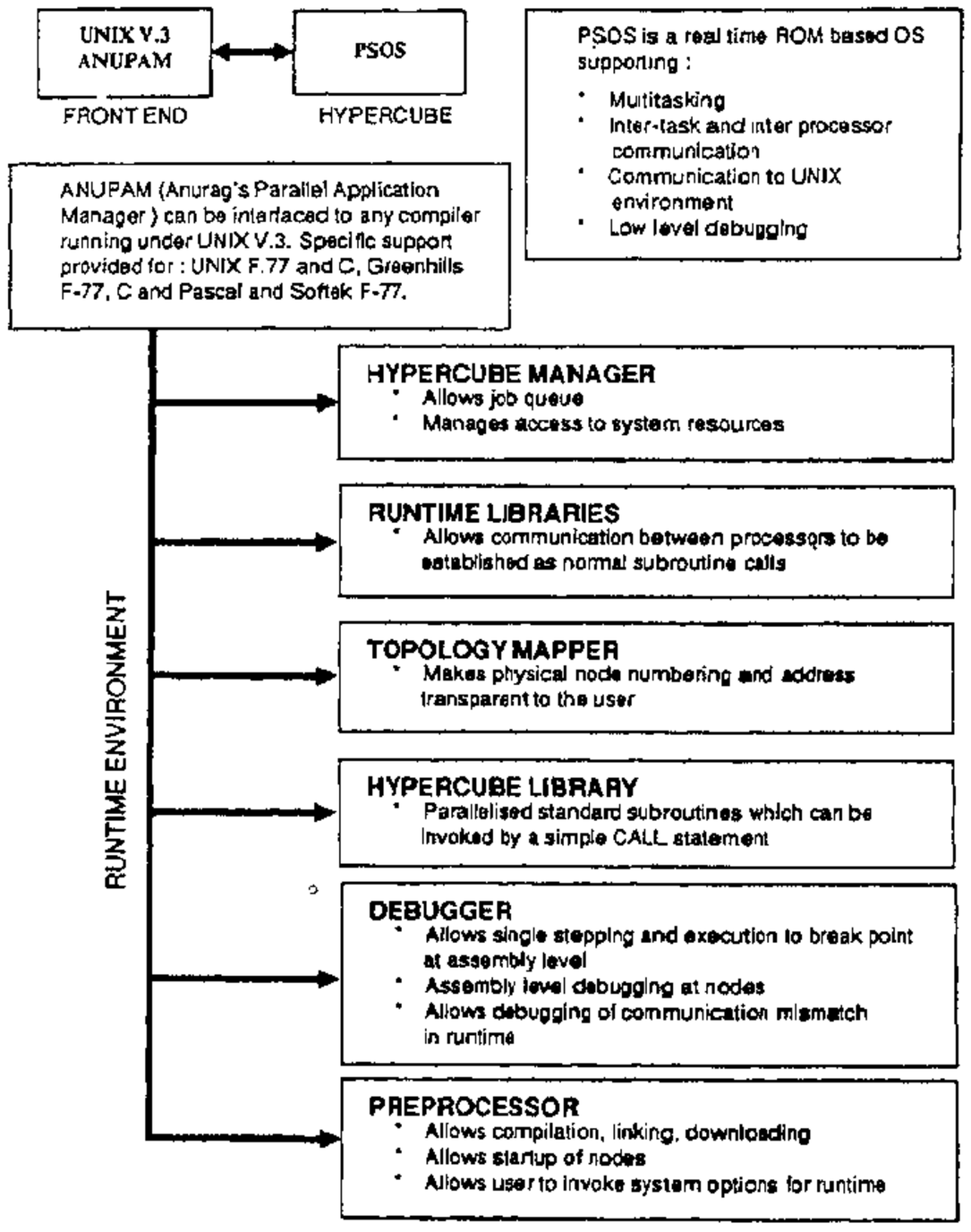

Figure 1. Flow chart showing the characteristics of the system software of PACE-8. The debugger shown in the figure indicates features currently available. Others are expected to be added.

$$S=\{T_{\text{node}}/T_{\text{par}}\}, \tag{1}$$

where T_{node} and T_{par} denote performance times for the same problem in the single-node and the multi-node systems respectively. Ideally, S would be equal to P, where P is the number of PEs (in the parallel machine), but in practice,

$$S=\eta P, \qquad \eta<1, \tag{2}$$

where η, the efficiency, is defined as

$$\eta=1/\{1+[T(\text{communication}/T(\text{computation})]\}. \tag{3}$$

If N denotes the number of floating-point operations in the problem and t is the time required per operation, then

$$T(\text{computation})=N\cdot t. \tag{4}$$

As for T(communication), if r packets are sent during the execution of the problem, each packet containing w words, then

$$T(\text{communication})=r\{t_{\text{startup}}+w\cdot t_{\text{send}}\}, \tag{5}$$

where t_{startup} represents the message startup overhead and t_{send} is the time to transmit one word. Using (4) and (5) in (3), we see that the efficiency η depends not only on the nature of the problem tackled but also on machine characteristics via the parameters t, t_{startup} and t_{send}. As should be evident from the above, the less the interprocessor communication needed, the closer would the speedup be to the ideal.

Turning to (5), in practice the messages communicated during the execution of a problem would not all have the same word length. Moreover, both t_{startup} and t_{send} would depend on word length. However, through the use of suitable average values $\langle t_{\text{startup}}\rangle$ and $\langle t_{\text{send}}\rangle$ for these quantities in (5), one can get an estimate for T(communication), which can then be used to obtain η and thence S.

Benchmarking

Computers (including those of the parallel variety) can be benchmarked in various ways. In our studies we have preferred to use the LINPACK program[6] developed by Dongarra of the Argonne National Laboratory, instead of Whetstone, Dhrystone, etc. LINPACK permits performance evaluation using a standard system of linear equations, and in a FORTRAN environment. It is pertinent to add that LINPACK involves arithmetical operations very representative of those occurring in large computer programs pertaining to scientific calculations.

The LINPACK program first generates a random, dense $(m\times m)$ matrix **A**, using a built-in random-number generator. The aim is to solve the system of linear equations.

$$\mathbf{A}\cdot\mathbf{x}=\mathbf{B}, \tag{6}$$

where x and **B** are column vectors, of which **B** is given. The **B** vector to be used in (6) is generated by computing **A**·**1**. Having generated **A** and **B**, the program then triangularizes the **A** matrix using the Gauss-elimination method. Pivoting is used to check for singularity and to keep the numerical errors small. The solution x is then obtained, and **A**·x is computed to see that (6) is satisfied. The time taken for triangularization and back substitution is obtained via a system call, and this time T is then used to obtain the MFLOP rating via the expression.

$$\{[(2/3)m^3+2m^2]/T\}, \tag{7}$$

where the numerator is the expression for the number of floating-point operations performed by the computer.

The essential feature of LINPACK is that it involves a high percentage of floating-point operations, making it an attractive touchstone. Indeed, not only does Dongarra report LINPACK ratings for a wide spectrum of computers, ranging from the CRAY to the IBM PC, but prospective buyers of advanced computer systems have also begun to pay serious attention to LINPACK performance, compelling manufacturers to quote LINPACK speeds in addition to theoretical peak speeds. It should be emphasized that while LINPACK exercises the computer very well, the LINPACK rating depends also on the operating system/compiler used. Further, for a given computer, improved LINPACK speeds can be achieved by using assembly versions of the Basic Linear Algebra Subprograms (BLAS)[7]. The results we report are not based on such coded BLAS.

Table 2 gives LINPACK speeds of a selection of computers. While many of the numbers have been taken out of a compilation by Dongarra[8], others are based on our own measurements using a program provided to us by Dongarra. Attention is called to the results for our nodal processors 68020/68881 (16 MHz) and 68030/68882 (25 MHz). While the results for the former were obtained with a SOFTEK compiler (supplied by Softek Private Ltd), the latter were measured using a GREENHILLS compiler (supplied by Greenhills Inc.).

The LINPACK program will not straightaway run on a parallel computer. Parallelization is required but here Dongarra permits flexibility. As he puts it: 'The manufacturer is allowed to use any algorithm to solve the problem. . . . The only restriction in running the benchmark program is that the driver program (supplied by the author of this paper) be run to ensure that the same problem is solved. The driver program verifies that the answer is correct and computes the

Table 2. LINPACK ratings (for problem size 100, double precision) of some computers.

System	Speed (MFLOP)	Performance time (sec)
CRAY XMP	66	0.0103
CDC CYBER 205	17	0.039
CRAY 1S	12	0.056
IBM 3090/150 E**	5.79	0.112
CDC 7600	3.3	0.210
NEC-S-1000/10**	2.83	0.242
ELXSI 6420	1.7	0.411
VAX 8650	0.7	0.98
CYBER 180/930 (Medha)**	0.58	1.18
ND 570	0.38	1.8
PACE-8 (Uniprocessor)**	0.22	3.12
IRIS 3130**	0.21	3.27
HONEYWELL-BULL DPS8/46**	0.192	3.562
TATA UNISYS**	0.165	4.145
MICROVAX	0.12	5.84
PACE-4 (Uniprocessor)**	0.064	10.7
VAX II/750	0.057	12.0
IBM PC/AT	0.0091	75.1

(i) Our measurements are indicated by a double asterisk.
(ii) The processor in PACE-4 is the 68020/68881 (16 MHz) combination and that in PACE-8 is the 68030/68882 (25 MHz) combination.

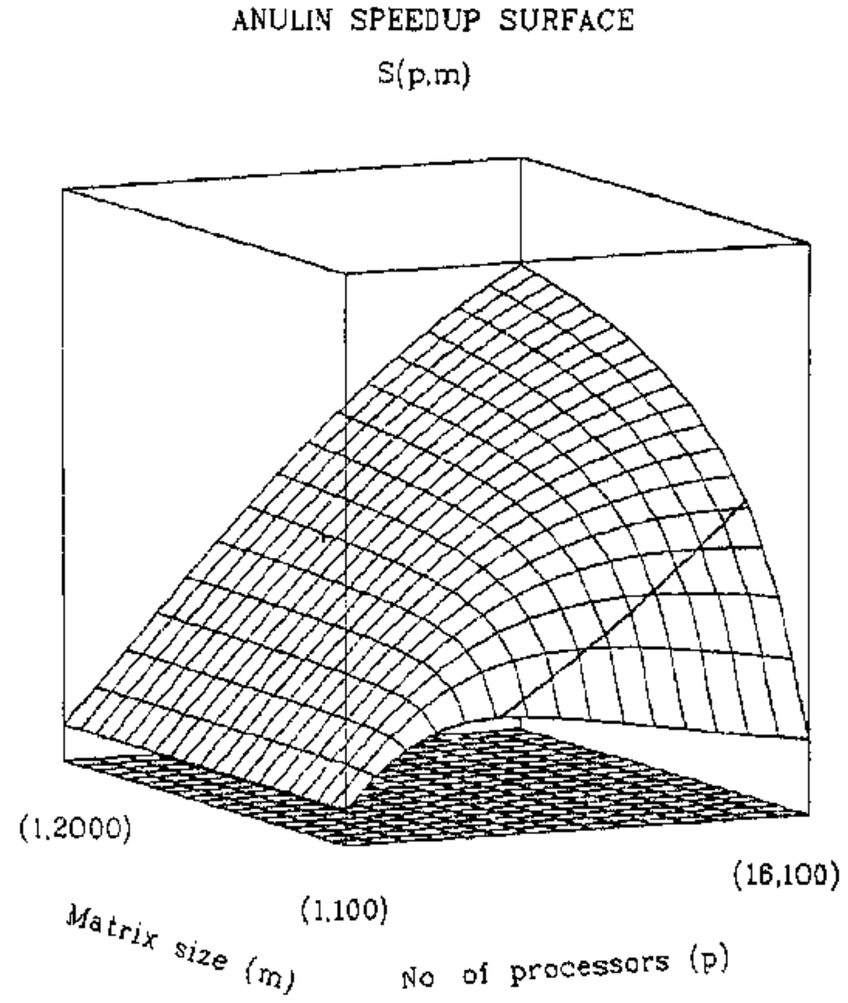

Figure 2. LINPACK speedup surface. For running this program on PACE we have developed a parallel version called ANULIN, consistent with the guidelines and constraints given by Dongarra. The surface above is based on ANULIN, and has been computed using (8) and values for $\langle t_{startup} \rangle$ and $\langle t_{send} \rangle$ appropriate to PACE-8. Absolute values of the speeds are not shown but they are easily established using the measured result for a uniprocessor. For a problem of a given size m, the LINPACK speed varies with the number of processors, becoming maximum at some value. The locus of this maximum is also shown.

total number of operations to solve the problem (independent of the method) $\{(2n^3/3)+2n^2\}$, where $n=1000$. (ref. 8).

In our experiments, we have used a parallel version of LINPACK developed by us (within the boundaries allowed) and referred to as ANULIN. Dongarra benchmarks all parallel machines using a system of linear equations of order 1000. Unfortunately, we were unable to run this problem on PACE-4 MK I and PACE-4 MK II on account of inadequate memory at the nodes. However, this deficiency has been made up in PACE-8. Table 3 gives a summary of the results obtained.

Figure 2 shows a three-dimensional plot of the LINPACK speed in a multiprocessor environment. It would be observed that, if the size of the matrix is fixed, the LINPACK speed first increases as a function of the number of processors, and then decreases. This is because communication dominates when the number of processors is small, degrading performance. On the other hand, when N is large the processors are underloaded. Thus, corresponding to each problem size, there is an optimum number of processors which gives the best speed. The locus of this maximum is also shown in Figure 2. The (1000×1000) problem chosen by Dongarra in the case of parallel computers appears to be a convenient compromise.

Table 3. LINPACK performance of various PACE models.

	PACE-4 MK I		PACE MK II		PACE-8 MK I	
	SP	DP	SP	DP	SP	DP
Uniprocessor	0.077	0.065	0.077	0.065	0.245	0.22
Multiprocessor	0.064	—	0.175	—	0.535	0.521

(i) All speeds in MFLOPS. SP, single precision; DP, double precision.
(ii) Problem size 100 in all cases except in PACE-4 MK I multiprocessor, where the size was 80.
(iii) Memory in PACE-4 MK I and MK II was too small to run the size 1000 problem.
(iv) Results obtained for size 1000 problem on PACE-8 are given in Table 4.

Comparison with other computers

One would naturally like to know how our system compares with other parallel computers. There have been several efforts in India to assemble parallel computers, of which FLOSOLVER[9] developed by the National Aeronautical Laboratory in Bangalore and PARAM recently announced by (C-DAC) the Centre for Development of Advanced Computing in Pune are perhaps the best known. Unfortunately, LINPACK speeds for these systems are not available. We have therefore restricted our comparison to Intel Corporation's iPSC system, using information published by Dongarra[8].

Like PACE, iPSC‡ also is a hypercube. Intel has come out with many versions, starting with the 80386/80387 combination for the PE and graduating to advanced ones where the coprocessor 80387 is replaced with more sophisticated floating-point accelerators such

as WEITEK 1167 and AMD vector processor. Considering that we have (as yet) not used any floating-point acceleration beyond that available with the coprocessor made by the manufacturer of the CPU, it is appropriate to compare PACE-8 with versions of iPSC not employing vector processor in place of 80387. Such a comparison is shown in Table 4. Obviously iPSC/2 performs slightly better, but iPSC/2 uses a direct-connect§ scheme for establishing communication, which leads to somewhat better speedups than we have at present.

Table 5 presents a summary of communication speeds pertaining to PACE-8 and the iPSC systems[10]. As already remarked, for purposes of estimating speedup one can use average values $\langle t_{\text{startup}} \rangle$ and $\langle t_{\text{send}} \rangle$ in (5). Using such average values for PACE-8, we estimate a speedup of ~7.51, which, taking the single-node rating for a problem size of 1000 as 0.26 MFLOPS (we observe that in going from a problem size of 100 to 1000, the single-node performance increases slightly from 0.24 MFLOPS to 0.26 MFLOPS), leads to a computed LINPACK rating of ~1.95 MFLOPS. The measured value (Table 4) is ~1.68. Considering the gross assumptions underlying (5), the agreement is as good as one might expect. A more detailed analysis which takes into account the details of the triangularization involved in LINPACK leads to the following ANULIN formula for η:

$$\eta = \frac{1}{1 + \dfrac{6P(P-1)\left[3\langle t_{\text{startup}}\rangle + \langle t_{\text{send}}\rangle (m+7)/2\right] + 6P\left(\langle t_{\text{startup}}\rangle + \langle t_{\text{send}}\rangle\right)}{t(m+1)(4m+11)}} \quad (8)$$

Use of this expression also leads to a speedup value of ~7.5. Formula (8) assumes no waiting time for the nodes whereas from detailed measurements we find that, of the 396 sec of performance time, computation and communication together account only for (333+24=) 357 sec. The balance presumably is waiting time. To check this out, more detailed studies are needed, which will be reported later. However, it is worth noting that if the measured values of T(computation) (333 sec) and T(communication) (24 sec) are used and the waiting time is ignored, then speedup S is ~7.46. which is what one would expect from (8).

Given our present communication scheme, some waiting is unavoidable. The direct-connect scheme reduces waiting. It is worth mentioning that, for large CFD problems, the speedup would be somewhat better than that predicted by LINPACK, since there would be more of computation and less of communication (and consequently, also, waiting for communication).

Several other studies besides LINPACK benchmarking have also been carried out by us on PACE-8, such as the Simplex problem, fast fourier transform, neural networks, fractals, molecular dynamics and aerodynamics. The results will be reported separately.

Table 4. Comparison of performance features of PACE and iPSC/2.

	PACE		iPSC	
No. of processors	Speed (MFLOPS)	Performance time (sec)	Speed (MFLOPS)	Performance time (sec)
2	0.42	1596	0.52	1280
4	0.80	832	1.03	652
8	1.68	396	1.99	337

(i) All results are for single precision; problem size 1000.
(ii) All PACE measurements done with the PACE-8 MK I system.
(iii) The iPSC/2 results are from reference 8.
(iv) The PACE clock is 25 MHz while the clock speed of the processor of the machine referred to by Dongarra is not known.
(v) The PACE-8 node is a 68030/68882 combination, whereas that for the node of the iPSC/2 machine benchmarked by Dongarra is not known. Circumstantial evidence obtained from various papers relating to iPSC systems on the one hand and WEITEK data sheets on the other suggest it could have been a 80386/WEITEK 1167 combination with a clock speed of 16 MHz.

Table 5. Latency and communication time per word for iPSC and PACE-8.

System	Message length	t_{startup} (μsec)	t_{send} (μsec)
iPSC/1	< 1 kbyte	1000	4
	> 1 kbyte	1000	7.5
iPSC/2	< 100 bytes	350	0.8
	> 100 bytes	660	1.44
PACE-8	< 100 bytes	49.57	6
	> 100 bytes (500 bytes)	45.3	6.04
	< 1 kbyte (800 bytes)	54.9	5.9864
	> 1 kbyte	49.97	6.1

(i) For PACE-8: $\langle t_{\text{startup}} \rangle \sim 50$ μsec, $\langle t_{\text{send}} \rangle \sim 6$ μsec/word.
(ii) Results for iPSC/1 and iPSC/2 taken from reference 9.

Future plans

Project PACE is being executed with hardware (system) support from the Electronics Corporation of India Ltd (ECIL). The CPU boards used so far are from their standard production line. However, keeping in mind the needs of PACE-128, ECIL are, on our suggestion, designing a more advanced version of their CPU board, which would also accommodate the special floating-point accelerator being designed by ANURAG. The use of this device, it is estimated, would result in very much improved LINPACK speeds.

Clearly, PACE-128 would need a more powerful and a more versatile FEP than we have at present. It has been our policy to concentrate on the hypercube and utilize for the FEP a uniprocessor that is locally available and compatible with our requirements. The FEP used in PACE-4 was ECIL's standard product UNIPOWER 20 and that in PACE-8 was UNIPOWER 30. The search is on for a suitable FEP with features

such as we consider desirable in the system to manage PACE-128.

The mechanical and electrical layout of a large multiprocessor system is a nontrivial task. Particular attention must be paid to ensuring absence of electromagnetic interference on the one hand and proper heat removal on the other. Simulation studies pertaining to these aspects are being jointly carried out by ANURAG and ECIL, in collaboration with the Centre for Electromagnetics of the Department of Electronics.

A crucial area needing attention is interprocessor communications. This becomes particularly necessary consequent to the significant enhancement in PE processing speed that would result with the use of our floating-point accelerator. The ANULIN surface corresponding to such processing speeds, but with existing communication speeds, is shown in Figure 3. As is evident, increasing the number of processors now results in diminishing returns in terms of speedup. However, this lacuna can be rectified, and a situation more like that in Figure 2 can be obtained by enhancing communication speeds. This is planned.

On the system software side, we intend to incorporate features that would permit a multi-user environment and a sharing of the hypercube. Provision for fault-tolerant operation in the event of a failure of one or more PEs is also being made. A simulator would also be loaded on the FEP to enable a prospective user to check out parallelization prior to execution on the hypercube. At present, an off-line simulator is available, capable of running on a PC/AT with a Xenix/UNIX operating system[11]. It has been our experience that, despite preliminary parallelization using the above simulator, problems are occasionally experienced while running jobs on the hypercube itself. To a large extent, this is due to difference between the compiler in our FEP and that in the simulator. Having an *in situ* simulator to carry a final checkout thus seems desirable.

Perhaps the most important improvement planned on the system software side is the symbolic debugger. While some primitive debugging facilities are already available, more features such as are likely to be desired by users are proposed to be added.

Preparations for the realization of the target machine are in full swing, and we hope it will be operational by end-1991. *En route*, intermediate versions, namely PACE-8 (MK II), PACE-16, PACE-32 and PACE-64, will be tried and tested. Meanwhile, scientists are welcome to try out problems of interest on the existing machine, i.e. PACE-8 MK I, and the soon-to-be-configured PACE-4 MK III, wherein the 68881 coprocessor of MK II will be replaced by the WEITEK board 3168. Considering that respectable LINPACK speeds are available even as it is (at least better than is commonly available in the country at present), we hope user interest will be widespread. It is not out of place to mention that the Caltech hypercube has been successfully utilized by a wide spectrum of scientists, ranging from high energy physicists to seismologists[1,12]. We hope that there will be a similar response from the Indian scientific community.

Note added in proof: After the manuscript was submitted, the system has been slightly tuned, improving the LINPACK rating from 1.68 to 1.84 MFLOPS.

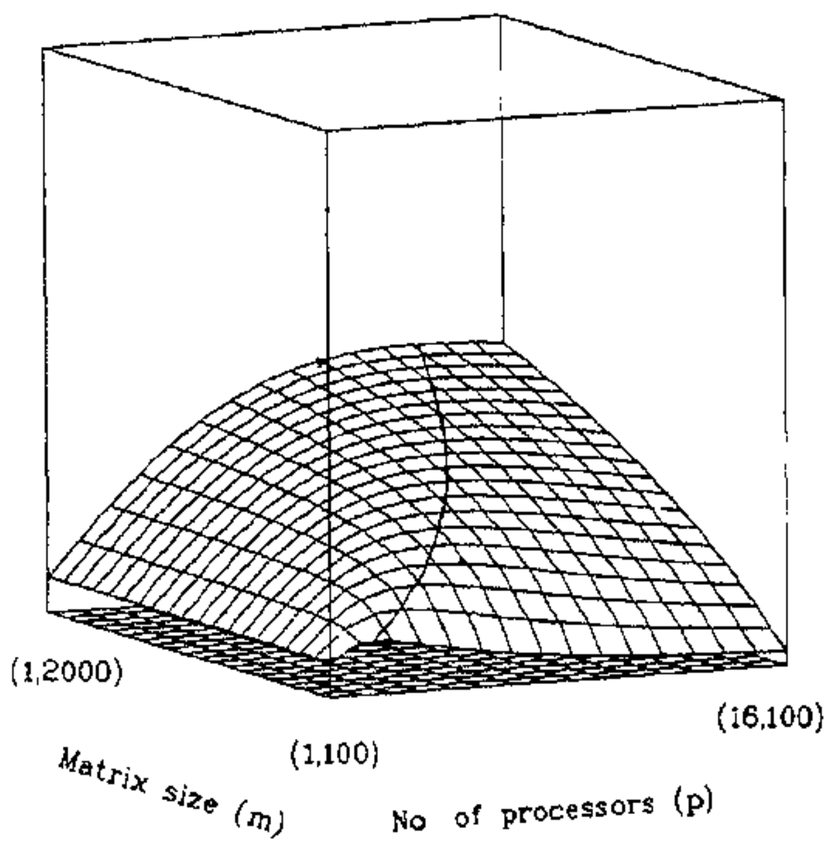

Figure 3. Surface similar to that in Figure 2 but corresponding to the situation where the time t for a floating-point operation is shortened, as would happen when our accelerator (now under design) is used. The communication speed is assumed to be the same as in the previous figure. Observe the striking difference compared to the previous figure. While the *absolute speed* would undoubtedly go up, speedup as defined in (1) is reduced for large P. To correct this deficiency, an improvement of communications is required, which is planned.

*UNIX is a trademark of AT&T Bell Laboratories.
†Ethernet is a trademark of Xerox Corporation.
≠iPSC is a trademark of Intel Corporation.
§Direct-connect is a trademark of Intel Corporation.
‖PSOS is a trademark of Software Components Group.

1. Fox, G., in *Supercomputers* (eds. Matsen, F. A. and Tajima, T.), University of Texas, Austin, 1986, p. 115.
2. Burns, P. *et al.*, in *Hypercube Concurrent Computers and Applications* (ed. Fox, G.), Association for Computing Machinery, 1988, vol. I, p. 872.
3. Venkataraman, G., *Phys. News*, 1989, **20**, 86.
4. Neelakantan, K. and Athithan, G., *PACE User Information*, ANURAG report ANU/PACE/89/01, 1989.
5. Fox, G. C., Johnson, M. A., Lyzenga, G. A., Otto, S. W., Salman, J. K. and Walker, D. W., *Solving Problems on Concurrent Processors*, Prentice-Hall International, New York, 1988, vol. I.
6. Dongarra, J. J., Bunch, J. R., Moler, C. B. and Stewart, G. W., *LINPACK User's Guide*, SIAM Publications, Philadelphia, 1979.

7. Lawson, C., Hanson, R., Kincaid, D. and Krogh, F., *ACM Trans. Math. Software*, 1979, **5**, 308.
8. Dongarra, J. J., *Performance of Various Computers Using Standard Linear Equations in a Fortran Environment*, Technical Memorandum No. 23, Argonne National Laboratory, 1989.
9. Sinha, U. N., Deshpande, M. D. and Sarasamma, V. R., *Curr. Sci.*, 1988, **57**, 1277.
10. Bomans, L. and Roose, D., *Concurrency: Practice and Experience*, 1989, **1**, 3.
11. Ghosh, P. P., Ganagi, M. S. and Ashok, A., *Simulator Users Manual*, ANURAG Report ANU/PACE/90/02, 1990.
12. Fox, G. and Otto, S., *Phys. Today*, May 1984.

ACKNOWLEDGEMENTS. We are deeply grateful to Dr V. S. Arunachalam, Scientific Adviser to Raksha Mantri, for enthusiastic support right from project conception. We are also thankful to Mr B. S. Prabhakar, Chairman and Managing Director, ECIL, for his unstinted co-operation, and to Dr J. Gopal Rao, Mr V. Guneshwara Rao and Mr S. A. Salaluddin of the R&D group of ECIL for active help at various stages. We also thank all our colleagues in ANURAG, who, in various ways, have made both the project and the present studies possible. G. V. thanks Dr J. J. Dongarra for making available the LINPACK program.

1 October 1990

RESEARCH COMMUNICATIONS

Effect of solar activity on the minor constituents in the mesosphere and lower thermosphere

T. S. N. Somayaji and T. Aruna Mani
Physics Department, Andhra University, Visakhapatnam 530 003, India

The effect of solar activity on the different minor species in the mesosphere and lower thermosphere has been studied using the one-dimensional model. The concentrations of all the species show significant increase with solar activity. The variations in the hydrogen and oxygen species are attributed to the variations in photodissociation coefficients of H_2O and O_2. The increase in the concentration of atomic nitrogen above 90 km is due to the ionic reactions in the lower thermosphere.

In the past decade studies of minor constituents in the middle atmosphere have acquired great importance, particularly with respect to the stratospheric ozone problem. Indeed, ozone is one of the many minor constituents produced by photochemical reactions in the middle atmosphere and its concentration is greatly affected through photochemical reactions involving other minor constituents. The UV region of the solar spectrum is known to vary over the 11 year solar cycle[1-3] and since the minor constituents in the middle atmosphere are produced mainly by the photochemical reactions that take place there, a solar cycle variability for the distributions of the minor constituents is expected. The solar cycle dependence of the distribution of minor species is important for investigations such as the composition, thermal structure and dynamics of the upper atmosphere and its state of ionization. Several sophisticated models of the mesosphere/lower thermosphere minor constituents have been developed and the variation of the distributions of the minor constituents arising from variations in solar illumination for different time scales have been studied earlier.[4, 5]

As a part of the Indian Middle Atmosphere Programme, we have developed a time-dependent one-dimensional model for the minor constituents in the mesosphere and lower thermosphere[6].

In view of the aforementioned importance of the solar cycle dependence of the distribution of minor constituents, we have applied this model for studying the variation of the distributions of the minor constituents from solar minimum ($R_z = 20$) to solar maximum ($R_z = 200$). The results of this study are presented and discussed in this paper.

The details of the computation technique were described earlier[6]. The solar irradiance and absorption cross-section data for the two solar activity conditions were taken from Deshpande and Mitra[7]. The vertical distribution of $O(^3P)$, $O(^1D)$, O_3, O_2 ($^1\Delta g$), H, OH, HO_2, NO, NO_2, $N(^4S)$ and $N(^2D)$ appropriate for the two levels of solar activity conditions were determined from the one-dimensional model.

The solar cycle dependence of the minor constituents is studied by the altitude distribution of a sensitivity parameter (S, as percentage variation) computed as

$$S = (Y_{max} - Y_{min})/Y_{min} \times 100$$

where Y_{max} and Y_{min} are the concentrations of the particular minor species for solar maximum and solar minimum respectively.

The variation of photodissociation coefficients (J) for H_2O, O_2 are shown in Figure 1. The solar cycle dependence of hydrogen (H, OH and HO_2), oxygen ($O(^3P, O\ D, O_3$ and $O_2^1 g$), nitrogen (NO, NO_2, N^4S and N^2D) species is shown in Figures 2–4 respectively.

Figure 2 shows that, below 90 km, the hydrogen species show a structure in the vertical distribution of their sensitivity to solar activity changes with three dominant maxima of about 45% at around 78, 86 and 90 km. The altitude region of these maxima agrees nearly with the region (70–85 km), where the photodissociation of H_2O has a maximum (as shown in